Problems We Had (Deliverable #7)

Our group had one main problem in converting our conceptual design into a working database. This problem could be explained in one word: inexperience. Essentially, our group agreed that this project would have gone much smoother if we had used SQL Developer previously.

The stress of uncertainty created many issues within our group. We were constantly disagreeing with each other. Eventually, this led to us trying to “divide and conquer.” Basically, we split up and worked on the database on separate machines. One particular group member insisted that he needed to enter data into the tables the second they were created. This created issues with adding attributes and creating relationships between entities. We believe we experienced a concurrency issue when this happened, but SQL Developer did not inform us of this. Not knowing what we were doing wrong, we were scouring the internet to figure out why the foreign keys and attributes we were adding were not being created. After several hours had passed, the group member left. By then, we were so convinced we did something wrong that we dropped every single one of our tables. We then re-created them and added the foreign keys, constantly checking the relational diagram to ensure it matched the one we had on paper. Eventually, we figured out what had happened, but a lot of time would have been saved if we knew what was going on.

Another issue we had was the unclear, vague errors that SQL gave us. A particularly frustrating one was ORA-02291, or “integrity constraint violated parent-key not found.” We spent a lot of time looking up this error code, but searching only resulted in more confusion. This is mostly because we had trouble finding a simple example on the internet; the SQL on Stack Overflow was far too advanced for us to even attempt to understand. Eventually, we discovered a feature in SQL Developer that allowed us to view constraints and keys. Through this feature, we discovered that SQL Developer had named our constraints very strangely because we were not aware that we had to name them ourselves. We fixed this issue by again, dropping and re-creating our tables and naming our constraints correctly. Another frustrating error we came across was ORA-00001, or “unique constraint violated.” We fixed this issue by deleting our constraints and re-creating them later. Essentially, if we had been more experienced and knew what these errors meant or if SQL Developer had explained them better, we could have avoided erasing our progress.

We also had another problem that was due to inexperience. Our scenario required us to have employees that were also customers. This would normally not be a problem, but employees and customers have different discount rates and participate in different reward systems. This led us to incorporate a subtype supertype relationship into our original E/R diagram. Later, we learned that we could not convert this into our database unless we learned the object-oriented capabilities of SQL Developer. Due to the upcoming project deadline, our group decided against doing this and came up with a different solution. We combined employee and customer into a table called “Person.” We then added all the employee and customer attributes into that table and allowed fields to be null so that we could follow the separate discount, promotion, and reward system requirements. We still don’t know anything about the object-oriented capabilities of SQL Developer, so we are not sure whether using it would’ve created a better solution. We do, however, know that we wouldn’t have had to change our E/R models, normalized relations, assumptions and data dictionary if we had used the object-oriented capabilities of SQL Developer.